How Does the System Know It's Right? Automated Confidence Assessment for Compliant Coding

by Yuankai Jiang, PhD; Michael Nossal, MA; and Philip Resnik, PhD

Abstract

Because there are numerous gray areas in code assignment, it can be difficult to define a single notion of correctness in medical coding. Beyond coding guidelines that may not be fully determinate, the desired outputs for a computer-assisted coding (CAC) system can vary depending on the consumer of the technology, the payer, the date of service, and other factors. Nonetheless, CAC systems need a well-defined basis for evaluating their own correctness in order to avoid presenting subpar coding results as trustworthy. In this paper, we present a solution to that problem.

Introduction

In any domain where computers are assisting in a human task, the issue of trust must be considered. Ironically, the better a technology works, the more likely people are to trust it, and therefore the more important it is for the system to assess accurately for itself whether or not its decisions require human review. Computer-assisted coding (CAC) illustrates this problem quite dramatically: When a system is trusted to submit automatically generated codes directly into billing, failing to flag questionable decisions for human review can lead to compliance failures that have serious financial and legal consequences.

The problem is made more complicated by the fact that "correct coding" decisions can be difficult to define. Databases of allowable CPT (procedure) and ICD (diagnosis) combinations can only help up to the point where context is needed. For example, such a database may tell you that an ICD code of 486 (pneumonia, organism unspecified) justifies billing for CPT 71010 (chest x-ray), but that choice may be inappropriate if the particular medical record indicates no evidence of pneumonia. Furthermore, coding guidelines can allow considerable room for variation,[1] and decisions about what should or should not be flagged for review may vary with the consumer of the technology, the payer, the date of service, and other factors.

In the face of this complexity, how can one establish a well-founded basis for trust in computer coding? This paper presents an answer based on combining rigorous statistical techniques—known variously as confidence estimation or confidence assessment—with unique attributes of CAC. In Section 2 we introduce the confidence assessment framework, and lay out a method of "situated confidence assessment" that utilizes coder-in-the-loop workflow in order to create a rigorous statistical model of how coders behave when reviewing automatically coded notes. In Sections 3 and 4, we present a large-scale experimental comparison of situated confidence assessment against alternative techniques, discussing "rules of thumb," databases of allowable CPT/ICD combinations, and context-independent conditional estimation based on CPT and ICD codes. Section 4 concludes with a summary of what we have learned.

Confidence Assessment

The Framework

Many computer systems can be viewed as making predictions. For example, speech recognition systems predict what words were said, based on the acoustic signal and the context, fraud detection systems predict whether a transaction requires investigation based on features including the transaction type, amount, and past history, and spam filtering systems predict whether the user does or does not want to see a given e-mail message, based on the content of the message.[2-4] The context for the present study is an engine that makes predictions (of procedure and diagnosis codes) for a medical note based on the metadata and language information present in the note.

In settings involving computer-assisted tasks, it is crucial for the computational system not only to make its best prediction, but also to adjust its behavior based on the extent to which its prediction can be considered reliable. For example, if a spam filter is not confident that its labeling of a particular message as spam is accurate, it should not silently move the message from the user's inbox to a spam folder without first giving the user the opportunity to review the decision.

This is a different problem from the prediction itself. Prediction can be viewed as seeking the choice C such that the conditional probability Pr(choice=C | evidence) is maximized, or the best choice to predict among all the possible alternatives, given the evidence. In contrast, the confidence one has in a prediction abstractly concerns the probability Pr(choice C is correct | evidence), or the probability that the yes-or-no answer to "Is C the right choice under these circumstances?" is yes. Methods for assessing that probability can be grouped as "confidence assessment" techniques and they play a role in many predictive settings.[5-7]

Consider an example. A (rather anthropomorphic) spam filtering program might ask itself, "Is this message more likely to be spam or not spam?", and it might choose to label the message spam based on a probability comparison of, say, .56 spam versus .44 not spam. But in order to assess the confidence of its choice, the question it must ask is: "When I label e-mails as spam for the particular set of reasons I used here, how likely is it that I am correct?" The difference between .56 and .44 may be very trustworthy in some cases, and not at all trustworthy in others. It depends a great deal on what evidence was used to arrive at those probability estimates.

Situated Confidence Assessment for Medical Coding

In the CAC context, the choice C is a set of codes for a medical note, and the behavior that depends on confidence assessment is whether to label an automatically coded note as confident and send it directly into a billing system, versus instigating human review of the autogenerated codes.[8] This description of the scenario applies equally well for rule-based coding systems, which can be viewed as predicting C with probability 1 when the rules assign C, and with probability 0 otherwise.

Human-in-the loop CAC workflow, however, provides opportunities for confidence assessment that would be difficult to exploit within a strictly rule-based framework. Every time engine output is reviewed by a human coder, the context and actions of that coder can be captured, creating an enormous repository of information about what codes are and are not likely to be approved without revision, under what circumstances. This makes it possible to model the CAC confidence assessment problem quantitatively in terms of predicting human actions: Pr(C=correct | evidence) can be interpreted as Pr(Approved | evidence), or the likelihood that a human coder will push the "approve" button given the engine's coding and the information in the note.

Formally speaking, our engine produces CPT (procedure) and ICD (diagnosis) codes, and every ICD code is linked to a single CPT code, and we can therefore view the engine's predicted codes as defining a set of vectors (c,i,Fc,Fi,V), where c is the CPT code, i is the ICD code, Fc is a subvector of evidence used in selecting the CPT code (i.e. maximizing Pr(c|Fc)), Fi is a subvector of evidence used in selecting the ICD code (i.e. maximizing Pr(i|Fi)), and V is a vector of other information available in (or created for) the particular (i,c) pair within the note.[9] We use the term *situated* confidence assessment to emphasize the

fact that the model makes use not only of c and i, but also features reflecting the context in which the coding took place.

CodeRyte's workflow includes human coders who review notes and either approve them without changes, or modify the codes. In the former case, approval means that every reported vector (c,i,Fc,Fi,V) involves a correct code pair (c,i), and therefore that approval action can be viewed as producing a "labeled" pair <(c,i,Fc,Fi,V),1>, where a label of 1 indicates "correct". Conversely, if the note was not approved without modification, then—regardless of the specific changes—the action can be viewed as producing a labeled pair <(c,i,Fc,Fi,V),0>, where a label of 0 indicates "incorrect."

Given a data set containing <(c,i,Fc,Fi,V),label> items, we train a binary classifier. Any of a huge variety of current supervised learning techniques—decision trees, support vector machines, maximum entropy models—can be used. Crucially, distinct classifiers can be built for distinct populations of notes and human coders; for example, it is straightforward to model the specific approval decisions associated with coding for a given hospital if so desired. Within any given model, different features may turn out to be important, and features range from obvious facts available in the metadata, such as the patient's gender, to information about the note, such as whether or not an Impressions section was present, to engine internal information about why the particular CPT-ICD combination was chosen, such as the document region where the ICD evidence was found or specific linguistic phrases that were present.

At runtime, the binary classifier computes a confidence in the range [0,1] for each input <(c,i,Fc,Fi,V)> corresponding to reported CPT and ICD codes of c and i, respectively, which can be interpreted as Pr(label=1|<(c,i,Fc,Fi,V)), or the probability that this CPT and ICD combination is correct. In order to combine evidence about multiple (c,i) combinations for a single note-level confidence estimate, we take the *minimum value* over Pr(label=1|<(c,i,Fc,Fi,V)) as the confidence value for the entire note. This is a very conservative strategy, in keeping with a primary focus on correctness and compliance: The entire set of codes for a note is considered only as strong as the weakest link. Every code reported in the note must pass the relevant confidence threshold (see discussion below) in order for the whole note to pass that threshold.

Crucially, the confidence assessment value has an intuitively clear interpretation. It is not just a heuristic value; rather, it can be viewed as a conservative estimate (because of the "weakest link" property) of the probability that a human coder (in the specific workflow associated with this customer site) would accept the entire note as correct without making any changes.

The behavior of a note in the human-in-the-loop workflow is determined based on a combination of rules and statistical confidence assessment. A first pass identifies clear-cut cases where human coding is necessary, e.g. when the note is missing crucial sections of dictation. A threshold for the note-level confidence assessment value is used to determine the behavior of the remaining notes: If the value is above threshold, then the note can be routed to a "confident" work queue primarily for quality assurance. The threshold is determined empirically, using a held-out data set, to ensure that for notes that pass threshold, CPT codes and primary ICD codes meet rigorous standards for correctness. Typically, the threshold is set so that in the notes that pass threshold, we obtain accuracies of 98 percent for CPT codes and 95 percent for primary ICDs. (See Jiang et al. 2006, for detailed discussion of CAC accuracy metrics.[10]) These requirements are as high or higher than the typical accuracies with which the coding industry is comfortable for human coders.[11]

Methods

We compared a number of techniques for identifying notes as "confident" based on coding by our NLP engine. Comparisons were done using a corpus of 39,279 radiology notes from a large billing company, coded by human coders using the CodeRyte coding interface to change or approve codes selected by the NLP engine. Following standard experimental practice in NLP, the total set of notes was divided into a training set (27,495 notes), a set used for development (786 notes), and a set kept separate and unexamined by the experimenters used for testing (10,998 notes).

Situated confidence assessment was considered along with the following techniques.

Rules of Thumb

A simple baseline technique we considered was the idea of establishing simple "rules of thumb" based on CPT codes. For example, one could imagine observing that both procedure and diagnosis codes for bilateral screening mammograms are generally coded well by the engine. Based on that observation, it would be straightforward to establish a rule of thumb that all procedures coded as 76092 could simply be presumed correct. A practical disadvantage of this technique, however, also makes it impervious to objective testing: it requires manually examining engine-coded notes and forming subjective judgments of which CPT codes are autocoded reliably. Under these circumstances, it is impossible to design an experiment disentangling the technique of "rule of thumb" creation from the individual or individuals creating the rules.

CPT/ICD Table

An alternative baseline, which can be tested rigorously, is to use a commercially available table specifying allowable combinations of CPT and ICD codes: Presume one can be confident in any CPT/ICD code combinations that appear in the table, independent of context. A whole note is considered confident if all the autocoded CPT/ICD combinations that appear in it are found in the table.[12] For this experiment we used 3M's CPT-to-ICD-9 Procedure to Diagnosis General Crosswalk.[13]

Nonsituated Confidence Assessment

Rather than using a database of allowable combinations, an alternative is to learn which combinations of CPT and ICD codes are reliable. This can be viewed formally as a variant of learning-based confidence assessment, which uses only the CPT and ICD code combination without benefit of contextual features.

When evaluating alternative techniques, the following measures are considered.

Capture Rate

When the technique is used to select which of the 10,998 test notes can be considered confident, what percentage of notes is selected?

CPT Correct

For notes identified as confident, what percentage of the CPT codes is correct, based on human coder judgments?

ICD-1 Correct

For notes identified as confident, what percentage of the primary ICD codes is correct, based on human coder judgments?

Results

Table 1 shows a summary of results on the test set. Confidence assessment using a table of CPT/ICD code combinations yields a capture rate of 58 percent. However, in that set, CPT and primary ICD accuracy both fall short of the target criteria, with a particularly large shortfall for primary ICDs.

For nonsituated confidence assessment, it is possible to vary the capture rate and accuracy values by adjusting the note-level threshold, as discussed in Section 2. Table 1 shows the results when the threshold is chosen to maximize capture rate while still ensuring that CPT and primary ICD codes meet the 98 percent and 95 percent accuracy criteria, respectively. If the primary ICD requirement is increased to 96 percent and the CPT accuracy to 99 percent, the capture rate for nonsituated confidence assessment decreases to 15.84 percent.

For situated confidence assessment, the threshold value is chosen in the same way. As the table shows, using situated confidence assessment makes it possible to more than double the number of notes achieving the 98 percent and 95 percent accuracy criteria, bringing the number of notes labeled confident close to 40 percent.[14]

Discussion

The results show that situated confidence assessment attains a capture rate of nearly 40 percent while maintaining CPT and primary ICD accuracy rates above 98 percent and 95 percent, respectively. As noted, these values are at least as rigorous as the performance typically expected of human coders. In contrast, using a commercial table of allowable CPT/ICD code combinations would permit far too many notes to be captured without achieving the high levels of accuracy needed to ensure trust in the automatic coding.

The comparison with nonsituated confidence assessment demonstrates the importance of context. When the full power of machine learning is brought to bear on a large set of contextual features determined by the specifics of the individual medical record, dramatic improvements are obtained.

In experiments involving supervised learning, it is standard to ask what the data requirements are in order for learning to be effective. This is particularly important in the CAC context, because it can take time to build up a sample of notes for which human review and correction/approval has taken place.

In order to address this question, we generated a learning curve showing the capture rate for varying quantities of training data, while always maintaining the rigorous criteria of 98 percent and 95 percent accuracy, respectively, for CPTs and primary ICDs. Figure 1 shows the capture rates for two smaller training set sizes (2,000 and 6,000 notes), one in the middle (12,000 notes), and three at the higher end (22,000, 25,000, and our full experiment training set of 27,495 notes). As the graph shows, a healthy capture rate of ~18 percent is obtained with only 2,000 training notes, with very rapid increases as more training notes are added. Note, of course, that these increases in capture rate take place while still maintaining the 98 percent and 95 percent accuracy criteria. (Note, too, that the engine achieves high accuracy even without confidence assessment based filtering.[15])

The pattern of results in our experiment would seem to make it clear that the "rule of thumb" approach is unlikely to be viable if high levels of CPT and primary ICD accuracy are required. The experiment establishes that more context is better. But formally, the rule of thumb is equivalent to the simple strategy of assigning the probability Pr(Approved|c) to 1 for CPTs c in some set of handpicked CPTs. This means decisions in a CPT "rule of thumb" approach would be based on less context than even the simplest and least effective of the methods we tested.

As an additional bit of data analysis, we looked at the distribution of CPT codes. Figure 2 shows the frequency of CPTs in the notes captured by situated confidence assessment in our experiment, ordered by rank. For example, in the test set of 10,998 notes, 1,111 notes (26.5 percent) out of 4,139 notes that were identified as confident include the most frequent CPT code, 71,010 (chest x-ray).[16] The shape of the curve, a Zipf distribution, is reassuringly familiar to those who frequently perform statistical analysis of naturally occurring data (Manning and Schuetze, 1999). It shows that the most frequent CPT codes in the confident notes appear quite frequently, but that the distribution has a long tail, or a very large number of distinct codes that occur infrequently.[17]

This distribution confirms that the results for situated confidence assessment are not based on singling out only a few very frequently occurring CPTs. Moreover, the long tail makes evident the importance of taking a statistical approach using large-scale data analysis. It seems quite unlikely that human coders or subject matter experts would detect patterns of reliability for codes appearing fewer than one time on average per 1,000 notes.

Conclusion

Situated confidence assessment combines statistical modeling with the unique character of human-in-the-loop coding workflow. A data-driven approach to system self-evaluation enables a clear, intuitive interpretation of confidence in coding output: The situated confidence assessment score for a note is a conservative estimate of the likelihood that members of the relevant population of human coders would approve the codes without making any modifications. The experiments in this paper demonstrate the accuracy of those estimates and therefore provide a rigorous basis for trust in computer-coding output.

Yuankai Jiang, PhD, is a Senior Software Engineer at CodeRyte, Inc., in Bethesda, MD.

Michael Nossal, MA, is a Senior NLP Engineer at CodeRyte, Inc., in Bethesda, MD.

Philip Resnik, PhD, is a Strategic Technology Advisor for CodeRyte, Inc., in Bethesda, MD, and an associate professor at the University of Maryland in the Department of Linguistics and the Institute for Advanced Computer Studies.

Notes

1. Nossal, Michael, et al. "Using Intrinsic and Extrinsic Metrics to Evaluate Accuracy and Facilitation in Computer Assisted Coding." Presented at the AHIMA/FORE Computer-assisted Coding Software Standards Workshop, Arlington, VA, September 2006.
2. Manning, C. and H. Schuetze. *Foundations of Statistical Natural Language Processing*. Cambridge, Massachusetts; London, England. MIT Press, 1999.
3. Bolton, Richard J. and David J. Hand. "Statistical Fraud Detection: A Review." *Statistical Science* vol.17, no. 3 (2002): 235-255.
4. Goodman, Joshua and David Heckerman. "Stopping Spam with Statistics." *Significance, the Magazine of the Royal Statistical Society*, vol. 1, no. 2 (2004): 69.
5. Blatz, J., et al. *Confidence Estimation for Machine Translations, Proc.* COLING-04. Geneva, Aug 23-27, 2004.
6. Delany, S.J., P. Cunningham, and D. Doyle. "Generating Estimates of Classification Confidence for a Case-based Spam Filter." Case-based Reasoning Research and Development, H. Munoz-Avila and F. Ricci (Editors), Springer Berlin/Heidelberg, 2005 177-190.
7. Hazen, T.J., S. Seneff, and J. Polifroni. "Recognition Confidence Scoring and Its Use in Speech Understanding Systems." *Computer Speech and Language* (16), 2002, 49-67.
8. Of course, in a real-world setting, even notes labeled "Confident" undergo random sampling and review for purposes of quality assurance.
9. Often a single CPT code will have multiple ICD codes associated with it, but not vice versa.
10. Jiang, Yuankai, et al. "How Does the System Know It's Right? Automated Confidence Assessment for Compliant Coding." *Computer-assisted Coding.* Presented at AHIMA/FORE Computer-assisted Coding Software Standards Workshop, Arlington, VA, September 2006.
11. One coding expert suggests that historically expectations in the 90- to 95-percent range have been typical, with recent increases as much as 95 to 98 percent. Anecdotally, however, it would appear that professionals in the coding field can have day-to-day expectations of their human coders that are significantly lower.
12. Formally, this can be viewed as a variant of confidence assessment that uses only the CPT and ICD code, and not other contextual features: The confidence estimate Pr(Approved|c,i) is 1 if this CPT/ICD pair (c,i) appears in the table and 0 otherwise.
13. The table contains 1.3 million entries and is marketed as "the most comprehensive library of coding links on the market today." It is derived from from "billing systems data, Medicare Carrier guidelines, AMA publications, and other public domain government sources" (http://www.data-files.com/cpt_icd.htm). One should note, of course, that our experiment is exploring using the table for a purpose other than the one for which it was designed.

14. With a threshold selected to approximately match the CPT/ICD table's 57.9 percent, situated CA achieves a capture rate of 58.17 percent with CPT correct rate of 97.03 percent and ICD-1 correct rate of 89.53 percent.
15. Stoner, Jean, et al. "Assessing Coder Change Rates as an Evaluation Metric." Presented at the AHIMA/FORE Computer-assisted Coding Software Standards Workshop, Arlington, VA, September 2006.
16. Since 71010 is the most frequent, we say it occurs at rank 1. The second most frequent code occurs at rank 2, and so on.
17. The lowest frequency in the graph is five occurrences. The dotted line in the figure shows the excellent fit of a Zipf curve to these data. The distribution of all CPTs in the 10998-note set is also clearly Zipfian. Zipf distributions characterize a huge variety of naturally occurring phenomena, ranging from word frequencies to the populations of cities.

Table 1

Confidence Assessment Technique Comparison

	Capture rate	CPT correct	ICD-1 correct
CPT/ICD table	57.9	96.62	83.15
Nonsituated CA	16.94	98.77	95.97
Situated CA	37.63	98.36	95.1

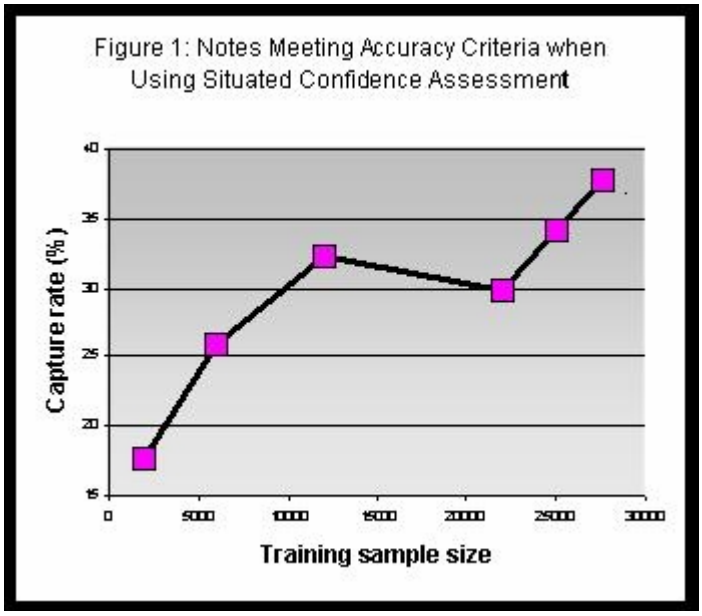
Figure 1: Notes Meeting Accuracy Criteria when Using Situated Confidence Assessment
Capture rate (%)
Training sample size
0
5000
10000
15000
20000
25000
30000

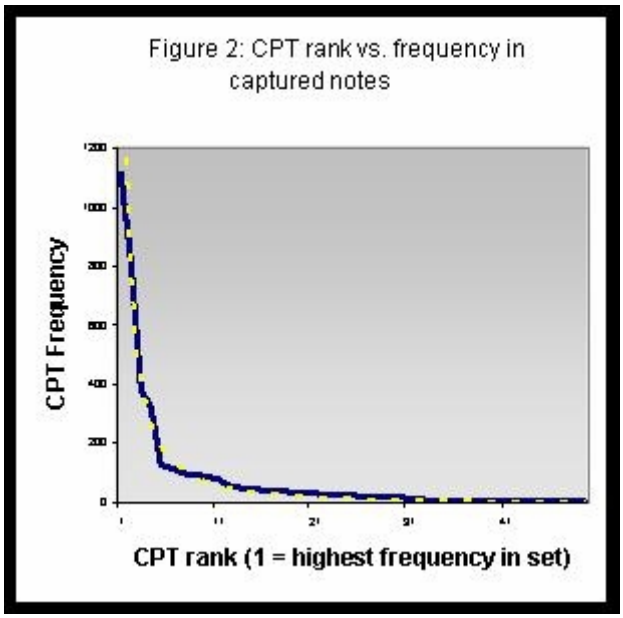
Figure 2: CPT rank vs. frequency in captured notes
CPT Frequency
CPT rank (1 = highest frequency in set)